斗鱼TV1.2.9插件系统文档20151109版本

1. **系统运行环境**

直播伴侣插件运行在WIN7、WIN8、WIN8.1、WIN10系统中，且插件应为32位Release版本。该插件系统针对直播伴侣1.2.9版本，不支持直播伴侣之前版本

1. **插件版本**

直播伴侣在加载插件时，首先会判断DLL中是否导出了GetPluginVersion导出函数，如果没有该导出函数，则加载插件失败，然后会调用该导出函数获取插件版本号，判断版本号是否等于PLUGIN\_VERSION，如何不相等则说明在开发的时候没有使用最新的plugin.h头文件，从而导致因版本不匹配导致插件加载失败。请下载最新的头文件版本进行重新编译

1. **插件入口函数**

IPlugin\* dyLoadPlugin(IPluginContext\* context)

该函数为插件DLL的导出函数，在该DLL中一定要导出该函数，否则直播伴侣将不会判定该DLL为直播伴侣插件。

参数：context为插件运行的上下文环境

返回值：返回为NULL则判定加载该插件失败

1. **IPluginContext接口**

该接口为插件运行环境接口，从dyLoadPlugin函数的参数中传入。

1. bool SendChatMessage(char\* msg)

发送聊天信息，参数为要发送的文本，采用GBK编码

1. bool GetUserName(char\* name)

获取直播伴侣中登陆的用户名，参数为待获取的用户名称，GBK编码

1. bool GetUserID(int\* id)

获取直播伴侣中登陆的用户ID

1. bool IsBroadcasting(int\* broad)

判断直播伴侣当前是否在直播，参数返回0为非直播状态，为1表示直播状态

1. **IPlugin接口**

该接口需要被使用者继承实现以下的两个接口函数，该接口作为dyLoadPlugin函数的返回值返回给直播伴侣，当返回值为空时，直播伴侣则将判断加载插件失败。

1. char\* GetName()

获取插件名称，采用GBK编码

1. IPluginElement\* CreateElement()

创建插件功能接口，如果返回空，则直播伴侣判定加载插件失败

1. **IPluginElement接口**

该接口需要被使用者继承试下部分接口，但是某些接口函数是必须要继承实现的。

1. bool Open(char\* env)

该接口必须被继承实现，参数env为通过调用GetCurrentStatus接口返回的字符串，用于保存直播伴侣退出后插件的配置信息。在使用该参数时一定要判断该参数是否为NULL。返回true时，表示插件插件成功，否则直播伴侣判定加载插件失败。

1. bool ReadVideo(uint8\_t\* pixels,int bufsize)

该接口为可选实现的接口函数。读取插件的图像，参数pixels为图像字节缓冲区，bufsize为需要拷贝的字节大小。

返回true时表示读取图像成功，可以在直播伴侣预览区域显示，否则读取图像失败。

1. bool ReadAudio(uint8\_t\* pcm,int bufsize)

该接口为可选实现的接口函数。读取插件的音频数据，参数pcm为音频字节缓冲区，bufsize为需要拷贝的字节大小。返回true时表示读取音频成功，可以在直播伴侣中播放该音频声音，否则读取音频失败。

1. bool Destroy()

该接口为可选实现的接口函数。对插件进行销毁。

1. bool GetParameters(PluginElementParam\* param)

该接口为可选实现的接口函数。该接口主要是获取图像和声音的参数信息，如要读取的图像的宽高，图像像素格式，或者音频的采样率，声道数等信息。如果要插件在直播伴侣上面显示图像或者播放声音，那么该接口函数必须要实现。返回true时可以成功获取图像或者音频的参数信息，否则不能显示图像和播放音频。

1. void SetConfig()

在直播伴侣设置功能中，会调用到此接口



1. void OnStartLive()

目前没有调用逻辑

1. void OnStopLive()

目前没有调用逻辑

1. void OnChatMessage(int id,char\* msg,char\* sendername,int deserveLev)

该接口为可选实现的接口函数。通知弹幕消息

参数id:发送者的用户id

参数msg:弹幕内容，GBK编码

参数sendername:发送者的昵称

参数 deserveLev:发送者的酬勤等级

1. void OnDonaterMessage(int id,char\* senderName,int sliver)

该接口为可选实现的接口函数。通知鱼丸消息

参数 id:发送者的用户id

参数 sendername:发送者的昵称

参数 sliver:赠送的鱼丸数量

1. void OnDeserveMessage(int id,char\* senderName,int num,int lev)

该接口为可选实现的接口函数。通知酬勤消息

参数 id:发送者的用户id

参数 senderName:发送者的昵称

参数 num:酬勤数量

参数 lev:发送者的酬勤等级

1. void OnUserEnterMessage(char\* senderName,int lev)

该接口为可选实现的接口函数。通知用户进入房间

参数senderName：进入房间的用户昵称

参数lev:获取用户的酬勤等级

1. char\* GetCurrentStatus()

该接口为可选实现的接口函数。获取当前的插件配置信息

返回值不为NULL，时直播伴侣会保存配置信息到数据库中，下次重新启动直播伴侣时，在该接口的Open函数中的参数env就为该接口返回的字符串。

1. bool OnCaptureData(int id,uint8\_t\* pixels[3],int pitch[3],int width,int height,PluginPixelFormat fmt,ImageSourceType type)

该接口为可选实现的接口函数。用于对视频图像的后处理。

参数 pixels:图像的像素。当为YUV类型的图像时，pixels[1]和pixels[2]可能为非空值，在使用时一定要判断数组是否为空

参数 pitch:图像每一个平面的字节行距。

参数 width:图像宽度

参数 height:图像高度

参数 fmt:图像像素类型，详见受支持的图像像素类型格式

参数type:图像源类型，详见ImageSourceType

1. void ShowPlugin(bool show)

该接口为可选实现的接口函数。控制是否显示插件

参数 show:为true时显示插件，false时不显示插件

1. void OnGiftMessage(int id,char\* sendername,char\* giftname,int giftnum)

该接口为可选实现的接口函数。通知鱼丸消息

参数 id:发送者的用户id

参数 sendername:发送者的昵称

参数 giftname:礼物名称

参数 giftnum:赠送的礼物数量

1. OnWindowMessage(HWND wnd,int msg,WPARAM wparam,LPARAM lparam)

分发windows消息

1. **像素格式**

插件系统中受支持的像素格式如下

enum PluginPixelFormat

{

PPF\_NONE,

PPF\_ARGB32, //B0 B1 B2 B3 B4 B5 B6 B7 G0 G1 G2 G3 G4 G5 G6 G7 R0 R1 R2 R3 R4 R5 R6 R7 A0 A1 A2 A3 A4 A5 A6 A7

PPF\_BGRA32, //R0 R1 R2 R3 R4 R5 R6 R7 G0 G1 G2 G3 G4 G5 G6 G7 B0 B1 B2 B3 B4 B5 B6 B7 A0 A1 A2 A3 A4 A5 A6 A7

PPF\_I420, //YUV 420 12bit 平面格式，分为Y、U、V三个平面 Y0 Y1 Y2 Y3 U0 U1 V0 V1

PPF\_NV12, //YUV 420 12bit 平面格式，分为Y、UV两个平面 Y0 Y1 Y2 Y3 U0V0 U1V1

PPF\_RGB565, //B0 B1 B2 B3 B4 G0 G1 G2 G3 G4 G5 R0 R1 R2 R3 R4

PPF\_RGB555, //B0 B1 B2 B3 B4 G0 G1 G2 G3 G4 R0 R1 R2 R3 R4 A0

PPF\_YUY2, //YUV 422 16bit 打包格式，只有一个平面 Y0U0Y1V0 Y2U1Y3V1

PPF\_UYVY, //YUV 422 16bit 打包格式，只有一个平面 U0Y0V0Y1 U1Y2V1Y3

PPF\_YVYU, //YUV 422 16bit 打包格式，只有一个平面 Y0V0Y1U0 Y2V1Y3U1

PPF\_YV12, //YUV 420 12bit 平面格式，分为Y、U、V三个平面 Y0 Y1 Y2 Y3 V0 V1 U0 U1

};

1. **图像源类型**

直播伴侣插件系统中受支持的图像源类型如下

enum ImageSourceType

{

IST\_CAMERA, //摄像头数据

IST\_IMAGE, //图片数据

IST\_VIDEO, //视频文件数据

IST\_PLUGIN, //插件图像数据

};

1. **音视频参数信息**

typedef struct \_PluginElementParam

{

int videoWidth; //图像宽度

int videoHeight; //图像高度

PluginPixelFormat pf; //像素格式

int pitch[3]; //行距

PluginAudioFormat af;

int audioSampleRate; //音频采样率

int audioChannels; //音频声道数

int audioFramesize; //一次采样数

int audioSound; //音量值

int pid //插件源在直播伴侣中对应的唯一id

}PluginElementParam;

1. **线程模型**

插件系统接口中，共有3个线程调用路径

|  |  |
| --- | --- |
| 线程 | 接口名 |
| douyuTool UI主线程 | |  | | --- | | Open | | SetConfig | | ShowPlugin | |
| 弹幕消息线程 | |  | | --- | | OnChatMessage | | OnDonaterMessage | | OnDeserveMessage | | OnUserEnterMessage | | OnGiftMessage | |
| 视频绘制主线程 | |  | | --- | | ReadVideo | | ReadAudio | | Destroy | | GetParameters | | GetCurrentStatus | | OnCaptureData | |

1. **运行插件**

将编译后的Release版本DLL复制到直播伴侣目录下的plugins文件夹里面，启动直播伴侣，就会搜索该文件夹下面的所有DLL文件，然后选择你开发的插件运行

1. **插件开发demo**

以下代码表示如何利用斗鱼插件系统开发直播伴侣插件。该demo的功能是实现简单的截屏功能

#define DYPLUGIN\_EXPORT

//包含插件头文件

#include "plugin.h"

#include <Windows.h>

class ScreenCaptureElement;

//继承实现IPluginElement接口

class ScreenCaptureElement : public IPluginElement

{

public:

ScreenCaptureElement()

{

HWND h = GetDesktopWindow();

HDC dc = GetDC(h);

m\_dc = CreateCompatibleDC(dc);

ReleaseDC(h,dc);

RECT rt;

::GetClientRect(h,&rt);

m\_width = rt.right;

m\_height = rt.bottom;

m\_size = rt.bottom \* rt.right \* 4;

BITMAPINFO bi;

memset(&bi,0,sizeof(bi));

BITMAPINFOHEADER &bih = bi.bmiHeader;

bih.biSize = sizeof(bih);

bih.biBitCount = 32;

bih.biWidth = rt.right;

bih.biHeight = -rt.bottom;

bih.biPlanes = 1;

m\_bitmap = CreateDIBSection(m\_dc, &bi, DIB\_RGB\_COLORS, (void\*\*)&m\_captureBits, NULL, 0);

m\_bitmapOld = (HBITMAP)SelectObject(m\_dc, m\_bitmap);

m\_show = true;

}

//open接口一定要重载，并返回true

virtual bool Open(char\* env){return true;}

virtual bool ReadVideo(uint8\_t\* pixels,int size)

{

if (m\_show)

{

HDC dc = GetDC(NULL);

BitBlt(m\_dc,0,0,m\_width,m\_height,dc,0,0,SRCCOPY);

ReleaseDC(NULL,dc);

memcpy(pixels,m\_captureBits,m\_size);

}

return true;

}

virtual bool Destroy()

{

DeleteObject(m\_bitmap);

return true;

}

//要显示截屏图像，一定要重载GetParameters接口，设置正确的图像参数，并返回true

virtual bool GetParameters(PluginElementParam\* param)

{

param->pf = PPF\_ARGB32;

param->pitch[0] = m\_width \* 4;

param->videoWidth = m\_width;

param->videoHeight = m\_height;

param->af = PAF\_NONE;

return true;

}

virtual void SetConfig(){}

virtual void ShowPlugin(bool show)

{

m\_show = show;

}

protected:

HDC m\_dc;

HBITMAP m\_bitmap,m\_bitmapOld;

uint8\_t\* m\_captureBits;

int m\_size;

int m\_width;

int m\_height;

bool m\_show;

};

//继承IPlugin接口

class ScreenCapturePlugin : public IPlugin

{

public:

//获取插件名称

virtual char\* GetName()

{

return "ScreenCapturePlugin";

}

//创建插件接口

virtual IPluginElement\* CreateElement()

{

ScreenCaptureElement\* elem = new ScreenCaptureElement();

return dynamic\_cast<IPluginElement\*>(elem);

}

};

//DLL导出函数

DYPLUGIN\_LOAD IPlugin\* dyLoadPlugin(IPluginContext\* context)

{

return dynamic\_cast<IPlugin\*>(new ScreenCapturePlugin);

}